

TM57 Series

TM57FA40&PA20&PA40 DEMO CODE
FOR TM57FA40&PA20&PA40
BASIC FUNCTIONS SAMPLE

Application Note

tenx reserves the right to change or discontinue the manual and online documentation to this product herein to improve reliability, function or design without further notice. tenx does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. tenx products are not designed, intended, or authorized for use in life support appliances, devices, or systems. If Buyer purchases or uses tenx products for any such unintended or unauthorized application, Buyer shall indemnify and hold tenx and its officers, employees, subsidiaries, affiliates and distributors harmless against all claims, cost, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use even if such claim alleges that tenx was negligent regarding the design or manufacture of the part.

AMENDMENT HISTORY

Version	Date	Description
V1.0	June, 2011	New release

CONTENTS

AMENDMENT HISTORY	2
PRODUCT NAME	4
TITLE	4
APPLICATION NOTE	4
01. ADC Application Sample	5
02. PWM Application Sample	6
03. Timer0 Application Sample	8
04. Timer1 Application Sample	9
05. WDT/WKT/XINT Application Sample.....	10
06. Buzzer Application Sample	12

PRODUCT NAME

TM57 Series IC

TITLE

TM57FA40 PA20/40_ADC_Application Sample

TM57FA40 PA20/40_PWM_Application Sample

TM57FA40 PA20/40_Timer0_Application Sample

TM57FA40 PA20/40_Timer1_Application Sample

TM57FA40 PA20/40_WKT/WDT/XINT_Application Sample

TM57FA40 PA20/40_Buzzer_Application Sample

APPLICATION NOTE

1. TM57PA20 and TM57PA40 have the same functions, the difference is TM57PA20 OTP ROM is 2K, doesn't support two times programmable, whereas, TM57PA40 OTP ROM is 4K, which supports two times programmable.
2. TM57PA40 and TM57FA40 have the same functions; however, some differences still do exist. Please refer to TM57 Series Differences Between TM57PA40 and TM57FA40 (AP-TM57PA40(FA40)_01EV10.pdf) which can be downloaded from <http://www.tenx.com.tw>

01. ADC Application Sample

1. Sample function: Test ADC0 (PA6) input voltage. The details of DEMO routine please refer to ADC.asm.

2. ADC related registers

F-Plane

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
05H	pad7	pad6	pad5	pad4	pad3	pad2	pad2	pad0
06H	pbd7	pbd6	pbd5	pbd4	pbd3	pbd2	pbd1	pbd0
07H	pdd7	pdd6	pdd5	pdd4	pdd3	pdd2	pdd1	pdd0
10H	adcdq11	adcdq10	adcdq9	adcdq8	adcdq7	adcdq6	adcdq5	adcdq4
11H	adcdq3	adcdq2	adcdq1	adcdq0	adcstart	adcsel2	adcsel1	adcsel0

R-Plane

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
05H	_	pa6	pa5	pa4	pa3	pa2	pa1	pa0
06H	_	_	_	_	_	_	pbe1	pbe0
07H	pde7	pde6	pde5	pde4	pde3	pde2	pde1	pde0
08H	_	npapu6	npapu5	npapu4	npapu3	npapu2	npapu1	npapu0
09H	_	_	_	_	_	_	npbpu1	npbpu0
0AH	npdpu7	npdpu6	npdpu5	npdpu4	npdpu3	npdpu2	npdpu1	npdpu0
0CH	buzen	adclks2	adclks1	adclks0	tm1psc3	tm1psc2	tm1psc1	tm1psc0
12H	nadcie7	nadcie6	nadcie5	nadcie4	nadcie3	nadcie2	nadcie1	nadcie0

3. Sample description

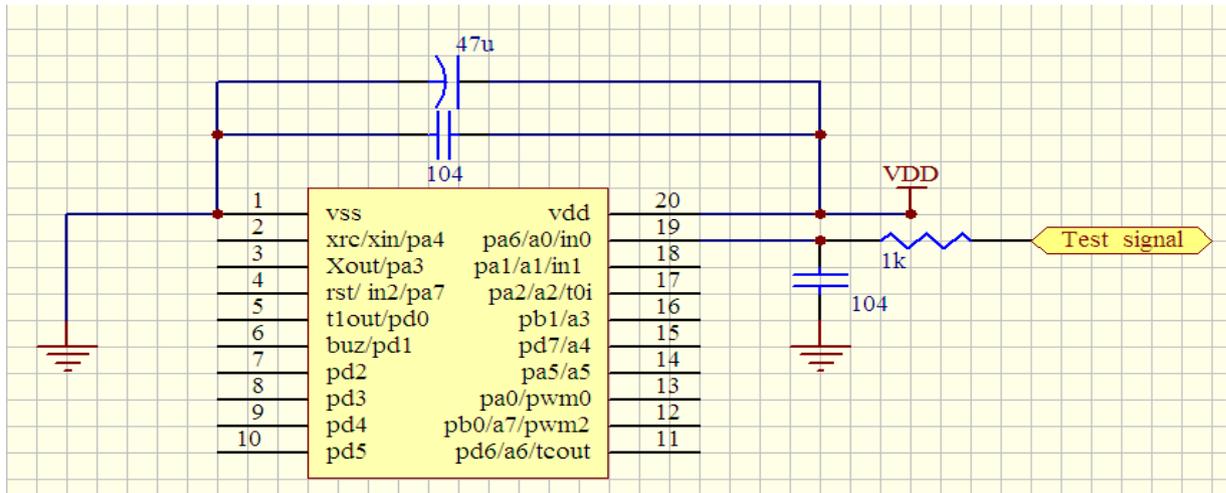
3-1. TM57FA40 PA20/40 has a 12-bit AD converter, the reference voltage is VDD. In application, AD port connects with an R, C low-pass filter, which is used to protect AD port and guarantee ADC accuracy, in general, R is 1 KΩ, and C is 104p.

3-2. AD converting steps:

- (1) Set AD converter port as input port, disable pull high, the corresponding I/O port data register is set to 1. For example, as in ADC0 setting:
pa6=0(R_plane,05.6);npapu6=1(R_plane,08.6); pad6=1(F_plane,05.6).
- (2) Select ADC conversion channel, i.e.: ADCSEL=0(F_plane,11h.2-0).
- (3) Set the I/O needed to ADC type, i.e.: nadcie0=0(R_plane,12h.0).
- (4) Set ADC conversion clock frequency (select proper ADC conversion clock frequency, the maximum is not more than 2 MHz). One AD conversion needs about 50 AD clock periods, i.e.: ADCLKS=1(R_plane,0ch.6-4)
- (5) Enable ADC conversion, i.e. ADCSTART=1(F_plane,11h.3). One AD conversion needs about 50 AD clock periods.
- (6) Wait ADC conversion finishes (after ADC conversion finishes, hardware will clear ADCSTART to 0 automatically).

- (7) ADC data handle. ADC result is saved in ADCDQ(F_plane,10h.7-0;F_plane,11h.7-4). Each time ADC conversion is finished, the result must be saved in the assigned location; otherwise, next time conversion result will overwrite the content in ADC data register.

4. Circuit diagram



02. PWM Application Sample

1. Sample function: After power on, PA0 output periodic cycle is 64 us, the duty is 1/2 waveform, PB0 output periodic cycle is 64 us, duty is 1/4 waveform. Details of the DEMO routine please refer to PWM.asm.

2. PWMA/B related registers

F-Plane

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0CH	pwm0duty9	pwm0duty8	pwm0duty7	pwm0duty6	pwm0duty5	pwm0duty4	pwm0duty3	pwm0duty2
0DH	pwm0duty1	pwm0duty0						
0EH	pwm1duty9	pwm1duty8	pwm1duty7	pwm1duty6	pwm1duty5	pwm1duty4	pwm1duty3	pwm1duty2
0FH	pwm1duty1	pwm1duty0						

R-Plane

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0BH	buzout	pwm0out	pwm1out	int1edge	tcout	tm1out	wktpsc1	wktpsc0

3. Sample description

3-1. PWMA/PWMB has similar structure; the clock source is system clock Fosc. When 8-bit counter equals to the 8-bit MSB of PWM DUTY register, PWM output signal will become

low level, the low 2-bit of LSB decides PWM output signal becomes high level immediately or after one periodic cycle delay.

Using 4 PWM periods as example:

PWMDUTY[1:0]=0, after each period overflows, it will delay one clock cycle before goes to high level.

PWMDUTY[1:0]=1, only the third period overflows, it will go to high level immediately.

PWMDUTY[1:0]=2, only the second and fourth periods overflow, it will go to high level immediately.

PWMDUTY[1:0]=3, only the second, third, fourth periods overflow, it will go to high level immediately.

High power level adds 0, 1, 2, 3 clock cycles respectively, therefore, PWM duty resolution can be expanded to 1024 clock cycles.

3-2. PWM output configuration steps:

- (1) Set PA0, PB0 ports as output, disable pull high.

```

pae0=1(R_plane,05.0);npapu0=1(R_plane,08.0);
pbe0=1(R_plane,06.0);npbpu0=1(R_plane,09.0)
    
```

- (2) Set duty of PWMA/B. PWM0DUTY(F_plane,0ch.7-0; F_plane,0dh.7-6), PWM1DUTY (F_plane,0eh.7-0; F_plane,0fh.7-6)

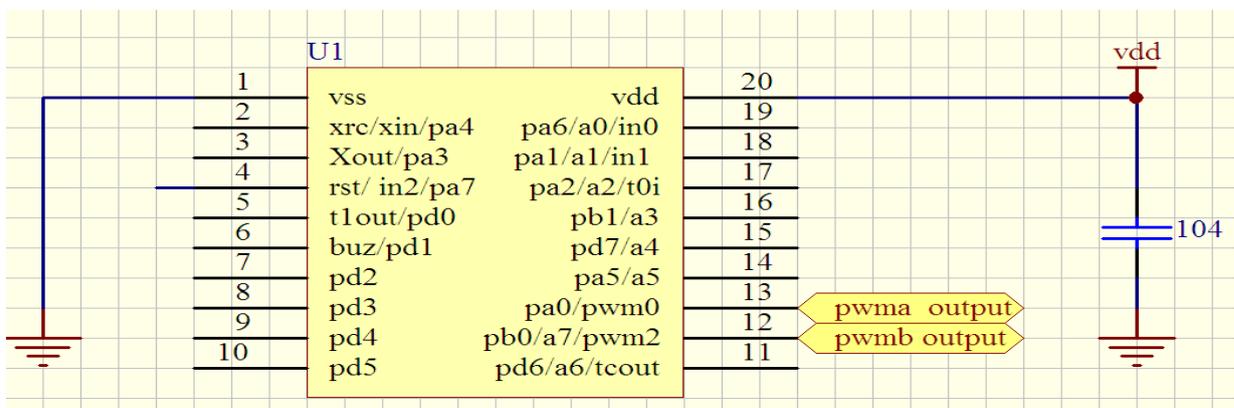
- (3) Enable PWMA/B output. pwm0out=1(R_plane,0bh.6),pwm1out=1(R_plane,0bh.5)

- (4) Disable PWMA/B output. pwm0out=0(R_plane,0bh.6),pwm1out=0(R_plane,0bh.5)

3-3. Clock cycle equation: Cycle T = (1/fosc)*256
Using routine setting value as sample: 1/4M*256=0.25 us*256=64 us

3-4. Duty equation: duty = PWMDUTYH/256+PWMDUTYL/1024
Example: PWMDUTYH=128, PWMDUTYL=2, therefore, duty=128/256+2/1024=514/1024

4. Circuit diagram



03. Timer0 Application Sample

1. Sample function: After power on, the LED output frequency is 500 Hz. Details of DEMO routine please refer to Timer0.asm.

2. Timer0 related registers

F-Plane

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
01H	Timer0 content							
09H	–	–	tm1i	tm0i	wkti	xint2i	xint1i	xint0i

R-Plane

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
02H	tcoutpsc1	tcoutpsc0	t0iedge	selt0i	tm0psc3	tm0psc2	tm0psc1	tm0psc0
0EH	–	–	tm1ie	tm0ie	wktie	xint2e	xint1e	xint0e

3. Sample description

3-1. Timer0 is an 8-bit increasing timer, when timer counts from 0xFF to 0, overflow occurs, timer interrupts. It doesn't support reload function, therefore, after interrupt, TIMER0 initial value needs to be reset. TIMER0 clock source option is decided by SELT01, if SELT01 = 0, the clock source is fosc/2, SELT01 clock source is input from T01 pin.

3-2. Timer time calculation equation:

$$\text{Time} = 1/\text{fosc} * 2 * \text{Timer0 Pre-Scale} * (256 - \text{TIMER0})$$

External counter calculation equation:

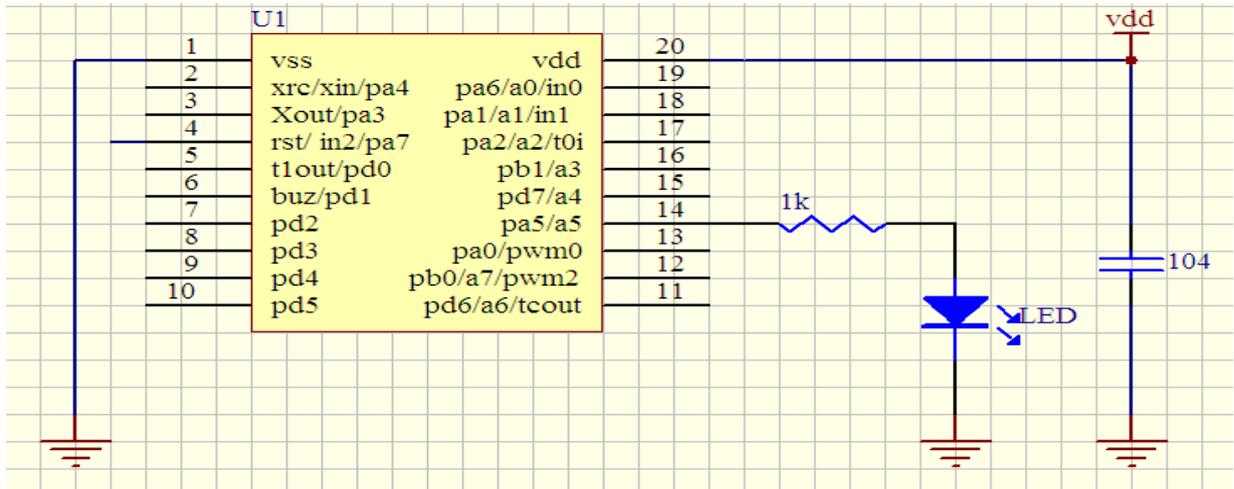
$$\text{Tcount} = 1/\text{Ftoi} * \text{Timer0 Pre-Scale} * (256 - \text{TIMER0}) \quad (\text{Ftoi is Toi input frequency})$$

Use DEMO as sample, timer 1 ms calculation is as follows: $1/4\text{M} * 2 * 16 * (256 - 131) = 1000 \text{ us}$, the time can be used for oscilloscope to observe LED light level changes.

3-3. Timer0 configuration steps:

- (1) Set TIMER0 clock source SELT0I(R_plane,02.4) and frequency division ratio TM0PSC(R_plane,02.3-0).
- (2) Set Timer0 initial value: timer0(F_plane,01.7-0).
- (3) Clear Timer0 interrupt flag tm0i=0(F_plane,09.4) and enable interrupt tm0ie=1(R_plane,0eh.4).

4. Circuit diagram



04. Timer1 Application Sample

1. Sample description: After power on, the LED output frequency is 500 Hz. Details of DEMO routine please refer to Timer1.asm.

2. Timer1 related registers

F-Plane

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0AH	Timer1 content							
09H	-	-	tm1i	tm0i	wkti	xint2i	xint1i	xint0i

R-Plane

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0CH	buzen	adclks2	adclks1	adclks0	tm1psc3	tm1psc2	tm1psc1	tm1psc0
0EH	-	-	tm1ie	tm0ie	wktie	xint2e	xint1e	xint0e
0DH	Timer1 reload offset value while it rolls over							

3. Sample description

3-1. Timer1 is an 8-bit auto-load timer, after setting initial value to timer1(F_plane,0ah.7-0) and tm1reload(R_plane,0dh.7-0), when timer overflows, it is not necessary to set timer initial value.

3-2. Timer1 clock source can only be FOSC/2, the time calculation is as follows:

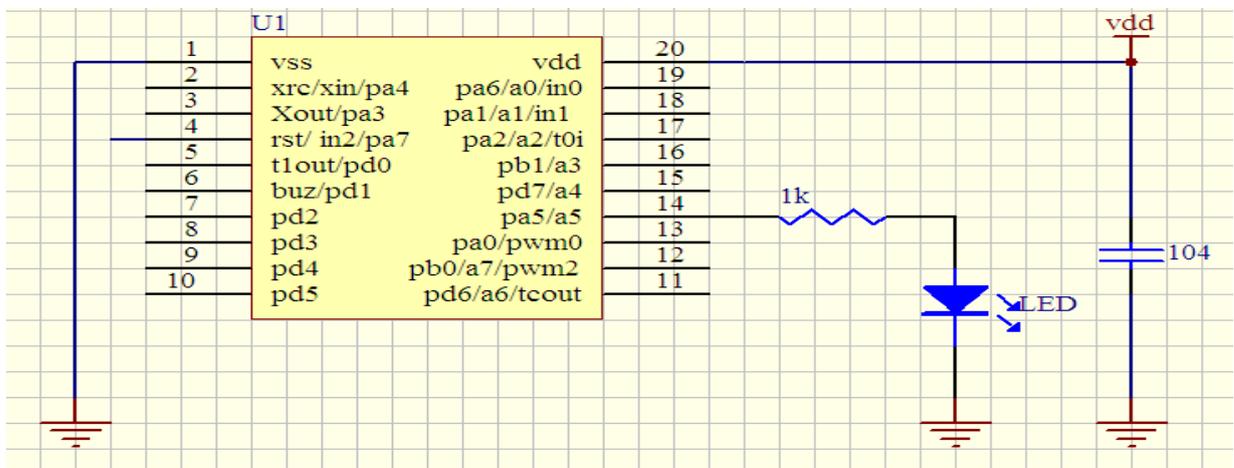
$$\text{Time} = 1/\text{fosc} * 2 * \text{Timer1 Pre-Scale} * (256 - \text{TIMER0})$$

Use DEMO as sample, timer 1 ms calculation is as follows: $1/4 * 2 * 16(256-131) = 1000 \text{ us}$, the time can be used for oscilloscope to observe LED light level changes.

3-3. Timer1 configuration steps:

- (1) Set TIMER1 frequency division ratio TM1PSC(R_plane,0ch.3-0).
- (2) Assign Timer1 initial value to timer1(F_plane,0ah.7-0) and tm1reload (R_plane,0dh.7-0).
- (3) Clear Timer1 interrupt flag tm0i=0(F_plane,09.5) and enable interrupt tm0ie=1 (R_palen,0eh.5).

4. Circuit diagram



05. WDT/WKT/XINT Application Sample

1. Sample function: under sleep mode, WKT and external interrupt is waken up from sleep mode, details of DEMO routine, please refer to WKT.asm, XINT.asm.

2. WDT/WKT/XINT related registers

F-Plane

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
03H	–	–	rambank	to	pd	z	dc	c
09H	–	–	tm1i	tm0i	wkti	xint2i	xint1i	xint0i

R-Plane

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0BH	buzout	pwm0out	pwm1out	int1edge	tcout	tm1out	wktpsc1	wktpsc0
0EH	–	–	tm1ie	tm0ie	wktie	xint2e	xint1e	xint0e

3. Sample description

3-1. When routine executing sleep, status register is PD=1, TO=0, it will enter power save mode. At this time, it can be waken up by reset (power on reset, low voltage reset, external pin (PA7) reset, watch dog timer reset) or external interrupt (3 external interrupts, WKT interrupt).

- (1) After power on reset, all system and external registers will reset to their default hardware setting.
- (2) Low voltage reset is set by SYSCFG Bit 11-10 LVR (LVR=11 is 2.1V, LVR=01 is 3.1V).
- (3) External pin reset is set by SYSCFG Bit 7 XRESETE (1: enable, 0: disable).
- (4) Watch dog timer reset is set by SYSCFG Bit6 WDTE (WDTE=0, watch dog timer disabled, WKT timer enabled; WDTE=1, watch dog timer enabled, WKT timer disabled). WKT and WDT are the same internal RC timer, in general mode and sleep mode, the operating condition is decided by WDTE and WKTIE, as following table:

WKTIE(R_plane,0eh.3) is WKT interrupt enabled.

Mode	WDTE	WKTIE	Internal RC Oscillator
General mode	0	0	Stop
	0	1	On going
	1	0	On going
	1	1	On going
Power save mode	0	0	Stop
	0	1	On going
	1	0	Stop
	1	1	On going

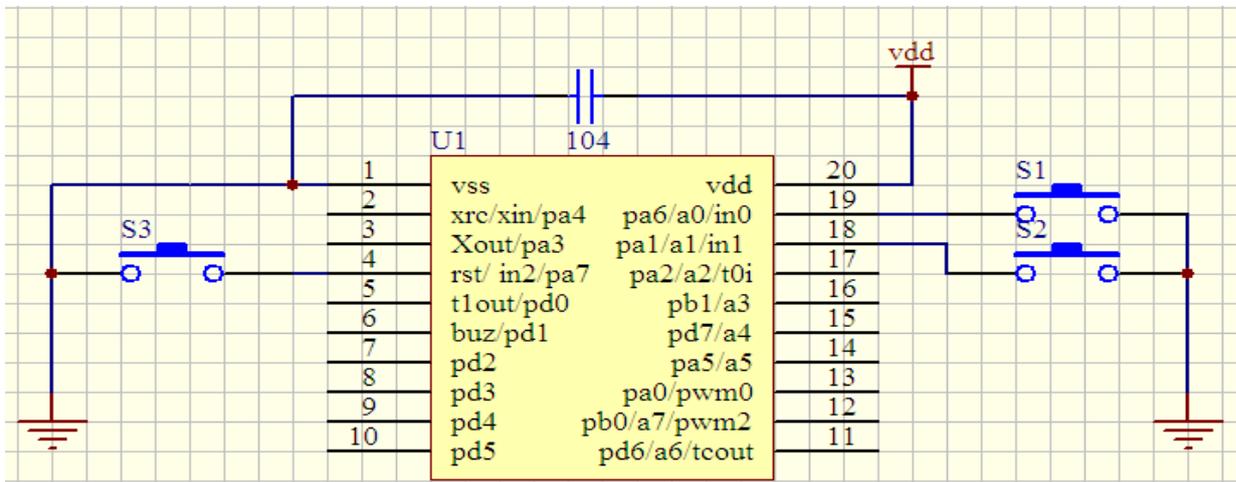
- (5) WDT reset wake up from SLEEP configuration steps: (refer to WDT.asm)
 - System sets register SYSCFG, WDTE = 1.
 - Set WDT overflow time, decided by WKTPSC(R_plane,0bh.1-0).
 - Enable WKTIE=1(R_plane,0eh.3) and execute SLEEP instruction to enter sleep mode.
 - Watch dog timer overflow resets from sleep mode to wake up.
- (6) WKT interrupt wake up from SLEEP configuration steps: (refer to WKT.asm)
 - System sets register SYSCFG, WDTE =0.
 - Set WDT overflow time, decided by WKTPSC(R_plane,0bh.1-0).
 - Clear WKT interrupt flag wkti=0(F_plane,09.3) and enable interrupt wktie=1(R_palen,0eh.3), then execute SLEEP instruction to enter sleep mode.
 - Watch dog timer overflows, interrupt occurs, sleep mode is waken up.

(7) External interrupt wake up from SLEEP configuration steps: (refer to XINT.asm)

- INT0 and INT2 can be triggered only by descend edge, in application, turn on internal pull high resistor and set as input; INT1 is bidirectional triggered, decided by int1edge(R_plane,0b.4) (0: descend edge, 1: rising edge). Set descend edge triggered, set as input port and turn on internal pull high resistor. Set descend edge triggered, set as input port, and turn off internal pull high resistor and connect pull low resistor.
- Clear external interrupt flag xint2i=0,xint1i=0,xint0i=0(F_plane,09.2-0) and enable interrupt xint2e,xint1e,xint0e(R_palen,0eh.2-0), execute SLEEP instruction to enter sleep mode.
- External interrupt pin changes from sleep mode to wake up, and enter external interrupt routine.

3-2. When IC is operating under general mode, WDT/WKT/XINT operating configuration steps are the same, except enter sleep mode.

4. Circuit diagram



06. Buzzer Application Sample

1. Sample function: PD1 pin output frequency 3.2K, details of DEMO routine please refer to Buzzer.asm.

2. Buzzer related registers

R-Plane

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0BH	buzout	pwm0out	pwm1out	int1edge	tcout	tm1out	wktpsc1	wktpsc0
0CH	buzen	adclks2	adclks1	adclks0	tm1psc3	tm1psc2	tpm1psc1	tmp1psc0
10H	buzpsc1	buzpsc0	buzprod5	buzprod4	buzprod3	buzprod2	buzprod1	buzprod0

3. Sample description

3-1. Buzzer includes 6-bit counter and time frequency divider, creates 50% duty square-wave. Set period and frequency division, then enable BUZOUT(R_plane,0bh.7), BUZEN(R_plane,0ch.7) to output the setting frequency.

3-2. Once buzzer counter overflows, buzzer output will be inverted once, therefore, the output frequency becomes half.

3-3. A clock cycle must be counted before turn off the buzzer output, at the same time, when the value of buzzer counter changes, the output frequency will change after next clock cycle.

3-4. Output frequency calculation equation:

$$F_{bz} = (f_{osc}/2)/(Instruction\ Cycle\ Divider)/(BUZ_PROD + 1)/2$$

3-5. Buzzer configuration steps: (please refer to Buzzer.asm)

- (1) Set buzzer to output and close pull high.
pde1=1(R_plane,07.1);npdpu1=1(R_plane,0ah.1)
- (2) Enable buzzer output to PD1 pin, i.e. buzout=1(R_plane,0bh.7).
- (3) Select buzzer clock frequency division BUZ_PSC(R_plane,10h.7-6) and buzzer period BUZ_PROD(R_plane,10h.5-0).
- (4) Allow buzzer counter output, i.e. buzen=1(R_plane,0ch.7).

4. Circuit diagram

